Workflow Step Reference

Levels Beyond
2710 Walnut Street | Denver, CO 80205 | (303) 495-2424
www.levelsbeyond.com

Table of Contents

Introduction

The **Reach Engine Workflow Engine** is responsible for all major operations from ingest through delivery of content and allows custom workflows to be created to override default functionality or expand on the functionality of the system.

The Reach Engine Workflow plugin abstracts basic tasks into individual modules that can be linked together with XML. Because the Workflow plugin knows nothing domain- or project-specific, the same code can be used to import a video into a Timeline in one project and into a Package in another, requiring only a few lines of project-specific XML to be written in each case, and creating individual modules with a high-level of reuse.

Additionally, existing workflows can be modified to add functionality to an existing process without deploying new code. For example, after you run the Reach Engine Ingest Asset workflow, you could run a distribution workflow.

Anatomy of a Workflow

A **Workflow element** provides general information about the workflow, including the workflow name, description, subject, and result information. Each workflow element must contain the following information:

- XSD and namespace definitions
- Workflow *id*

Steps define the actions the workflow should take. Each workflow must have an `<initialStepName>` element as the first child element in the workflow. The *initialStepName* must refer to a valid step in the workflow. Each step must contain:

- A *step type* that is referenced via the xml namespace/schemaLocation in the workflow element.
- A *name* to identify the step within the workflow.

Transitions are within steps and define which step should be run next. Transitions must contain:

- A *condition*: An expression that is evaluated when a step completes. If true, the workflow goes to the step defined in `targetStepName`.

  Note: If a workflow step fails to complete, none of its transitions are evaluated, and workflow execution stops.
- A *targetStepName:* Defines the workflow step to perform next when a condition is true.

Context Data Defs (contextDataDef) are like variables for workflows, and are used to accept external input to your workflow, maintain and communicate data between steps and sub-flows, and store the result of a workflow. Context Data Def values are determined at runtime and must contain:

- A *datatype* that defines the type of data.
- A *name* that is used to reference the contextDataDef.

For more information about workflows, see the Levels Beyond Workflow Guide.

Creating Custom Workflows

Levels Beyond provides an XML schema that you can use to create your own workflows The XML document references multiple XML schema and the schema declaration calls the schema from the 'latest' schema location: http://www.levelsbeyond.com/schema/latest/studio.xsd.

Workflow Element

The workflow element declares general information about the workflow itself, including a unique key, name, descriptive information, subject information, and workflow result information. For example:

```
<workflow xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://levelsbeyond.com/schema/workflow"
    xmlns:nimbus="http://levelsbeyond.com/schema/workflow/nimbus"
    xsi:schemaLocation="http://levelsbeyond.com/schema/workflow http://
www.levelsbeyond.com/schema/latest/studio.xsd"

    id="sendTimelineToFtp"
    name="Send Timeline to FTP"
    description=""
    subjectDOClassName="Timeline"
    deadlineExpression=""
    sdkVersion="5.0"
    >…
</workflow>
```

In the basic workflow element, Workflow XML is based on XML Schema Documents (XSDs) provided by Levels Beyond. XSDs provide a validation structure for your XML. If you have a good XML editor, such as Oxygen XML Developer, the editor can read the supplied XSDs and provide instant validation and auto-completion.

Note: The workflow XSDs are updated periodically as new versions of Reach Engine Workflow are released. When a new version is available, version numbers can be updated.

Workflows are interesting because they can interact with files and send emails, but they are *useful* because they can also interact with Reach Engine DataObjects. The most common way for a workflow to interact with a data object is through its subject. A workflow's subject is defined, in generic terms, in the *subjectDOClassName* attribute of the `<workflow />` tag.

After you set up the header attributes, you can add Reach Engine Workflow elements to your Workflow XML document.

Attributes

id required STRING	A unique identifier for this workflow within your Reach Engine Studio UI installation. When you import a Workflow XML, Reach Engine Studio UI looks at the *id* attribute value and does one of two things: If a workflow already exists in the system with the same ID, it creates a new version of that workflow with your XML. Otherwise, it creates a new workflow in the system. Using IDs allows you to modify workflows over time to correct a problem or introduce new functionality.
name required STRING	Specifies the name of the workflow as it will appear throughout the Reach Engine Studio User Interface (UI).
sdkVersion required STRING	The version for this workflow schema.
description STRING	A more verbose description of what the workflow does. This information is used primarily as a reference when looking at the workflow through an XML editor.
executionLabelExpression STRING	Describes an individual execution of your workflow. A *Workflow* is really a *definition* of a workflow, while a *Workflow Execution* is the result of running a workflow. The *executionLabelExpression* is a label for a single run of the workflow within Reach Engine and is evaluated each time that workflow is run. The text entered here displays in logs as well as in the Workflow tab of the Reach Engine UI. Expressions may be added here to make the values that display unique each time the workflow runs, based on the subject of the workflow.
showInUserInterface BOOLEAN	Declares whether this workflow should display in the Reach Engine Studio UI. The default value is true, which means that the workflow displays as an action that users can take (“true”). However, you may want to hide your workflow from a user, for example, if your workflow includes other workflows that should only be triggered by a schedule or watchfolder, or you have workflows that only exist as *sub-flows*.
subjectDOClassName required STRING	Specifies which kind of data objects this workflow should be available to execute against. All data in Reach Engine, whether an asset, a clip, a project, or a user, is represented by a Data Object. Specify a *subjectDOClassName* to ensure: 1. The Reach Engine validates that an appropriate Data Object is provided as the “subject” when your workflow starts. 2. The Reach Engine Studio UI only displays your workflow against the appropriate Data Objects (if the showInUserInterface boolean is true). For example, if you declare your *subjectDOClassName* as “Clip”, then your workflow only displays after one or more Clips are selected in the UI. If no value is specified, then the workflow applies globally.
resultDataDef STRING	Specifies the name of a *contextDataDef* that represents the result of the workflow. Workflow results can be interpreted by the UI for display to the end user, and can be aggregated by the Execute Subflow Step to represent the result of the step.
devWorkflow BOOLEAN	Used to define dev workflows. Dev workflows are filtered from search results for status screens and the like. Default value is false.

group STRING	Specifies a workflow group to which this workflow should be associated. Reach Engine will automatically create the group if it does not already exist. When selecting workflows in the UI, group-based workflows will all be listed under the group for easy organization.
deadlineExpression STRING	Unused at this time.
subjectQualifierExpression STRING	Expression that, if set, will validate passed in subjects. The qualifier expression must resolve to a Boolean. The root context of the expression is the subject Data Object.
adminOnly BOOLEAN	States whether this workflow should only be viewable/executable by admin users. Default value is false.
defaultExecutionMode _WORKFLOWEXECUTIONMODE	States how this workflow should run when executed from the Reach Engine UI. By default, workflows run asynchronously, meaning that when a user starts a workflow execution, it will execute in the background. However, by setting this value to “sync”, when the user starts a workflow execution the UI will wait until the execution completes. Default value is async.

Tip

When you tell Reach Engine Studio UI to run a workflow with a given ID, it always uses the latest version for the workflow execution.

Currently, you cannot go back in the version history.

Elements

initialStepName required	Required element that must be the first child element in the workflow. Defines the initial step this workflow should start when the workflow is run.
allowedRole	Specifies a Reach Engine role name that should have access to this workflow for execution. One or more roles may be specified by creating multiple allowedRole elements. If no allowedRole elements are specified, this workflow is available for execution by all users in the system.
abstractWorkflowStep	Refers to the abstractWorkflowStep element.
CONTEXTDATADEF	Context Data Defs provide you with workflow variables that can be used to accept external input to your workflow, maintain and communicate data between steps and sub-flows, store the result of a workflow, and have a default value based on the fields of DataObject.

Common Attributes and Elements

The following attributes and elements are used within many of the workflow elements.

Attributes

name required STRING	Specifies the name of the step. A step's name must be unique within the workflow definition. It is used as a key for all other workflow data that refers to a step, like initialStepName and targetStepName.
displayName STRING	Specifies a display name for this workflow step that is used in Reach Engine workflow status user interfaces. If omitted, the name of the step is used.
devStep BOOLEAN	If true, this step is not displayed in Reach Engine workflow status user interfaces. Default is false.
executionLabelExpression STRING	Specifies a label expression for this workflow step. The execution label expression is evaluated at the time that the step is executed and can contain execution-specific contextual information, like the resolved value of context data defs.
continueOnException BOOLEAN	If true, an exception that occurs during step processing does not cause the workflow to stall. Instead, the error is logged but the workflow continues. A true value is often used for steps deemed inconsequential to successful workflow processing, like sending an FYI-type email. Default is false.
priorityExpression INTEGER	Expression that should resolve to an integer to determine priority order if this step qualifies for a workflow queue. If queues are not being used, this expression has no effect. Default is 0.
pctComplete INTEGER	States the percentage of this workflow that has been completed when this step is done. Many steps can leverage this information with the percent complete upon starting the step to provide incremental percent complete updates during step execution.
stepContextExpression DATA OBJECT	Advanced property that provides the workflow designer with the ability to add strong DataObject contextual information to a workflow step. This allows custom Reach Engine workflow status UIs to easily display context data to a particular workflow step execution. This expression must resolve to a DataObject class that extends AbstractWorkflowStepContext.

Common Elements

contextDataDef

Context data definitions are user-definable variables. They are the primary way of communicating values to workflows and between steps and subflows. They can be set explicitly when starting a workflow execution and/or can have a default value expression set upon them.

Context Data Defs provide you with workflow variables that can be used to accept external input to your workflow, maintain and communicate data between steps and sub-flows, store the result of a workflow, and have a default value based on the fields of DataObject.

Context Data Defs are created by specifying a name and a data type. Other attributes provide you with control over how the contextDataDef will be used and an optional default value expression. To conform to the Workflow XSD, they must be the last elements in your workflow.

Context Data Defs are defined just above the closing `</workflow>` tag.

Attributes

name required STRING	Specifies the name for this dataDef. The name must be unique within the workflow definition. The name is used throughout the workflow to call this value and can be references in expressions, like ${exampleDataDef}.
label STRING	Specifies a label that should be displayed in user interfaces that allow for dynamic workflow executions. This label provides the UI code with a user-readable label to display when dynamically populating an input form.
dataType required CONTEXT DATA TYPE	Defines the kind of variable (i.e., data type) for the contextDataDef. Each type has unique capabilities and should be applied judiciously. Further, step definition properties usually require certain data type(s) as input values, so a particular property can only be supplied by a dataDef of a certain type.
defaultDataExpression STRING	Defines the default value for this dataDef if one is not supplied initially. If a defaultDataExpression does not resolve to a value (null), it will be re-evaluated until it resolves to a value. If the contextDataDef is given a value from a user or config parameter during the workflow initialization, that value overrides the result of this expression. **Note**: A defaultDataExpression element is also available that allows for easier, multi-line string editing. If the contextDataDef contains content in both the defaultDataExpression element and attribute, preference is given to the attribute.
multiple BOOLEAN	Specifies whether this data def contains multiple values or not. It provides the underlying data storage for the contextDataDef and typically contains an array or collection of values, hold strings, or files. If a dataDef is set as single (multiple=false) and multiple values are presented to it, only the first value is retained. If a dataDef is set as multiple (multip0le=true) and a single value is presented to it, it will work as a collection of values with only one element. Default value is false.
userInput BOOLEAN	Specifies whether this data def should be presented to users in the Reach Engine UI as an input parameter. A dataDef is presented to the user if userInput=true or if required=true and no *defaultDataExpression* is defined. Default value is false.
subjectDataPath STRING	If the workflow has a subject, setting subjectDataPath causes the workflow engine to pass the value of this dataDef into the subject DataObject via the specified path.
required BOOLEAN	Marking a dataDef as required means that a workflow cannot run without this dataDef having a supplied value. If a *defaultDataExpression* is provided, that expression must resolve to a not-null value. Otherwise, a value must be provided to the workflow at the time of workflow execution. Default value is false.

Elements

defaultDataExpression	An alternative form to the defaultDataExpression attribute. This element allows for easier multi-line string editing. Preference is given to the attribute if both forms have content.
description	Supplies a description of this data def for workflow documentation purposes.

picklist	Defines a list of valid values for this data def. Values can come from either a metadata property of type picklist or multilist or from a declared list of picklistItem elements inside this picklist.

Example Usage

```
<contextDataDef name="transcodedFile" dataType="File"/>
<contextDataDef name="transcodedFileName" dataType="String" userInput=”true”
defaultDataExpression=”${transcodedFile?.name}”>
<picklist>
  <picklistItem value="${transcodedFile?.name}" label="File Name"/>
  <picklistItem value="${transcodedFile?.name}_${#getDay(date)}"
label="Filename with Date"/>
</picklist>
</contextDataDef>
```

exception-transition

Exception transitions provide the workflow author with the ability to transition workflows based on specific exceptions that may arise from a step. If not provided, any exception causes the workflow to stall.

Attribute

type required STRING	Specifies the exception type to match.

Element

type required	Specifies the name of the target step of this transition. The name must match the name of a step in this workflow or else the workflow definition will be rejected upon import.

transition

Transitions provide the workflow engine with one or more conditions that will dictate the next step to execute. Transition conditions are evaluated in order, and the first one to resolve to true will have its target step executed.

Attribute

condition required STRING	Specifies the condition expression for this transition. The expression must resolve to a true/false value.

Element

targetStepName required	Specifies the name of the target step of this transition. The name must match the name of a step in this workflow or else the workflow definition will be rejected upon import.

fallbackTransition, retryFallbackTransition, stallFallbackTransition, waitFallbackTransition

Allows the user to customize error handling for steps.

nextStep

If this step will not have multiple transitions and will only go to the next step, use this property to define the best step to transition to.

Custom Workflow Components

The table below gives a brief description of the elements available in the Workflow schema.

Elements	
addPickListItemStep	Creates new picklist item to incrementally manage picklist population in workflows.
checkinAsset	Checks in a locked asset. It is very similar to the Save Asset step except that it also deals with asset locking.
compressFileStep	Compresses one or more files into a ZIP format.
confirmWorkflowDataStep	Emits a resolvable workflow event to users. The user confirms or overrides the value of the configured workflow variable and then resolves the event.
convertAudioStep	Converts an audio file or asset into a target format with trimming and pre/postroll options.
convertClosedCaptioningStep	If MacCaption integration is enabled in Reach Engine, the Convert Closed Captioning step provides facilities to convert and manipulate CC files.
convertImageStep	Converts an image asset or file into a different image format using ImageMagick.
convertVideoStep	Converts a video file or asset into a target format with trimming and pre/postroll options.
copyFileStep	Copies a file or string content to a target directory.
createCsvStep	Generates a CSV file from the result of a DataObject query with select parts. By using select parts, a tabular result is created.
delayStep	Puts the workflow execution into a WAITING status until the configured delay time/deadline has been reached.
deleteAssetStep	Deletes the specified asset from the Reach Engine database and from the repository. It should only be used if you truly want to remove all traces of the asset from the system, such as in the case of a bad import.
deleteDataObjectStep	Deletes data from the Reach Engine database.
deleteFileStep	Permanently deletes one or more files and/or directories. Note that this delete is not reversible.
emailStep	Produces email text with optional attachments and sends it to the email addresses specified.
embedClosedCaptioningStep	If MacCaption/Manzanita integration is enabled in Reach Engine, the Embed Closed Captioning step provides facilities to embed closed captioning into movie files.
enableWatchFolder	Ensures that there is an enabled watch folder at the specified location and config.
episodeConvertAudioStep	Extends the Convert Audio step with Telestream Episode Engine-specific parameters.
episodeConvertVideoStep	Extends the Convert Video step with Telestream Episode Engine-specific parameters.

executeSubflowStep	Provides workflows with the ability to call other workflows (called subflows) and pass related data down to those subflows. A subflow execution is created for each related data item that is selected. These subflows all execute in parallel, but the executeSubflowStep waits until all of these subflows complete before continuing to its transitions.
failWorkflowStep	Puts the workflow execution into a FAILED state. Unlike a stalled workflow, a failed workflow cannot be resumed.
getVideoConversionResult	Attempts to obtain a result from a video conversion started with the *startVideoConversion* step.
groovyStep	Executes a groovy script.
logStep	Logs the *outputExpression* result to the Reach Engine log files.
noopStep	These steps do not do anything; they move to transitions. No-op steps are useful for workflows where the first "real" step must be based upon a transition evaluation.
queryStep	Queries the Reach Engine database for data objects.
raiseWorkflowEventStep	Raises a Workflow Event of the type specified. Workflow Events are a special way to message users about events that occur during workflow executions. Users can subscribe to event types and configure how they want to be notified when an event occurs.
readXmpMetadataStep	
reviewAndApproveStep	Emits a resolvable workflow event to users. The user reviews the material attached to the event and either approves or rejects it.
rhozetConvertVideoStep	Extends the Convert Video step with Rhozet Carbon-specific parameters.
runCommandStep	Runs external processes either locally on the host Reach Engine server or remotely via a Reach Engine Remote Exec server.
saveAssetStep	Creates or updates an asset in Reach Engine's repository.
saveDataObjectStep	Saves new or modified data to the Reach Engine database.
sendFileStep	Sends any file to a remote location.
setContextData	Sets the specified contextDataDef to the value generated by *valueExpression*.
startVideoConversion	Launches a video conversion and returns an ID that can be used to look up the result later in the workflow. This step is different from the Convert Video step in that it does not wait for the conversion to complete before marking the step completed.
subflowContextDataMapping	Subflow context data mappings allow for other data defs to be passed down to the subflows. The data types of the two data defs must match.
testStep	Do not use.
updateDataObjectCalculatedPropertiesStep	Triggers an update of a DataObject's calculated properties.
validateXmlStep	Validates an XML file's syntax for well-formed-ness.

Using Custom Workflow Components

The workflow components in this section can be used to customize your workflow.

addPickListItemStep

Creates new picklist item to incrementally manage picklist population in workflows.

This step contains the following default variables:

- continueOnException
- devStep
- displayName
- executionLabelExpression
- pctComplete
- priorityExpression
- stepContextExpression

See common attributes and elements for details.

This step contains the following step-specific variables.

Step-Specific Variables

propertyName STRING	Specifies the name of the metadata property to add a picklist item to (must be a picklist type).
pickListLabel STRING	Specifies the label of the picklist item to add.
pickListValue STRING	Specifies the value of the picklist item to add.

Example Usage

```
<addPickListItemStep name="addItem"
    propertyName="${myField}"
    pickListLabel="${myLabel}"
    pickListValue="${myValue}">
     <transition condition="${true}">
        <targetStepName>go somewhere</targetStepName>
    </transition>
</addPickListItemStep>
```

checkinAsset

The Checkin Asset step checks in a locked asset. It is very similar to the Save Asset step except that it also deals with asset locking.

This step contains the following default variables:

- continueOnException
- devStep
- displayName
- executionLabelExpression
- pctComplete
- priorityExpression
- stepContextExpression

See common attributes and elements for details.

Step-Specific Variables

assetExpression STRING	Expression resolving to an asset that is used as the target of the asset save. The expression must resolve to an asset.
contentExpression required STRING	Expression resolving to the location of the binary content to save into the repository. The expression must resolve to either a file or a valid string file path that is the location to which the binary content is saved in the repository.
contentTemplateExpression STRING	Specifies the content template that should be assigned to the saved Asset. Content templates provide a "profile" to the content that is stored against an asset. Assets can contain multiple contents; therefore, you can save content against an asset with template "Foo" and other content against the same asset with template "Bar", then retrieve each content separately later. It is possible to specify content without a template, but if you try to save the asset again later with other content and no template, the new content will effectively overwrite the previous content. The expression must resolve to a valid string template name or a content template DataObject.
checkinDescriptionExpression STRING	Description of the changes that prompted this asset check-in. The expression must resolve to a string.
keepLockedExpression STRING	Expression that states whether to keep the lock on the asset or not. Must resolve to a Boolean. Default is false.
resultDataDef STRING	Stores the result of the asset check-in into a dataDef. The dataDef must be a DataObject.

Example Usage

```
<checkinAsset contentExpression="/Content/movie1" name="checkin"
contentTemplateExpression="movie_template">
</checkinAsset>
```

compressFileStep

The Compress File step compresses one or more files into a ZIP format.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	An expression that should resolve to one or more files and/or directories. All items are compressed into a single file. If a directory is specified, all directory contents are compressed. The resulting compressed file is created in the parent directory of the first item specified, and is made available via the resultDataDef context variable.
targetDirectoryExpression STRING	If specified, the compressed file is created in the directory path that is the result of evaluating the expression. Otherwise, the step creates the compressed file in the parent directory of the first source file. The expression must resolve to a directory or a string that is a valid directory path.
targetFilenameExpression STRING	If specified, the compressed filename is the result of evaluating this expression. Otherwise, the file name is the compressed file the same as the first source file, but with an appropriate extension.
resultDataDef STRING	Stores the compressed file into a dataDef. The dataDef must be a file.

Example Usage

```
<compressFileStep sourceFilesExpression="Content" name="compressFolder"
targetFilenameExpression="backup"></compressFileStep>
```

confirmWorkflowDataStep

The Confirm Workflow Data step emits a resolvable workflow event to users. The user confirms or overrides the value of the configured workflow variable and then resolves the event.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

confirmationDataDef required STRING	The name of the contextDataDef to confirm/override.

resolutionDataDef STRING	The name of the contextDataDef that holds the result of the resolution. This value is not the actual context data value, but rather the expression of what the user did. Therefore, this value is null until resolution is performed, which allows future steps to block on this variable until it is resolved.
waitForResolution BOOLEAN	Declares whether or not the workflow execution should wait until the emitted workflow event is resolved before continuing. Default value is true. If false, the workflow will continue processing.

convertAudioStep

The Convert Audio step converts an audio file or asset into a target format with trimming and pre/postroll options.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	Defines the source audio file to convert. The expression must resolve to a file, an audio asset, an audio asset content, or a string that represents a valid audio file path.
targetContentTemplateExpression STRING	An expression that represents the target format of the audio conversion. The expression must resolve to an audio content template or the name of a valid audio content template.
mediaConversionTemplateExpression STRING	An optional expression that represents the target audio conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template you should fully qualify it by specifying the converter type as well as the template name (e.g., "Vantage:MyWorkflow").
inputChannelCountExpression STRING	Defines the number of input channels for channel mapping.
outputChannelCountExpression STRING	Defines the number of output channels for channel mapping.
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated audio. The prerollFileExpression must resolve to an audio asset, an audio file, or a string that represents a valid audio file path.
prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.

postrollFileExpression STRING	Includes a postroll to the generated audio. The postrollFileExpression must resolve to an audio asset, an audio file, or a string that represents a valid audio file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.
trimStartExpression STRING	Trims from the start of the main audio resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main audio that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.
trimEndExpression STRING	Trims from the end of the main audio resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main audio that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
resultDataDef STRING	Stores the converted audio file into a dataDef. The dataDef must be a file.

convertClosedCaptioningStep

If MacCaption integration is enabled in Reach Engine, the Convert Closed Captioning step provides facilities to convert and manipulate CC files.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	An expression that must resolve to the source CC file to convert. The expression must resolve to a file, an asset, an asset content, or a string representing a valid file path, and the target file must be a valid closed captioning file.
targetCcFormatExpression required STRING	An expression that represents the target format of the CC conversion. The expression must resolve to a ClosedCaption Content Template, a target extension, a target CC template name, or a valid MacCaption type key.

timecodeOffsetExpression STRING	An expression that, if specified, causes the result CC file to have its timecodes rippled by the offset amount. The offset is in decimal-seconds format, so the expression must resolve to a double. A negative number means to subtract seconds off the source timecode. For example, if the workflow should ripple all timecodes up by one hour, then the timecodeOffsetExpression should be 3600.
sourceVideoFrameRateExpression required STRING	In order to correctly do timecode manipulation, the source video's framerate is required. This expression should evaluate into a double value.
sourceVideoTimecodeFormatExpression STRING	In order to correctly do timecode manipulation, the source video's timecode format is required. The expression must resolve into a string that denotes whether the timecodes are in Drop Frame or not. Valid values are "Drop Frame", "DF", "Non-Drop Frame", or "NDF".
trimStartExpression STRING	You can trim the CC conversion result starting at the first caption after the specified trim start. The trimStartExpression must resolve to a double representing the number of seconds into the video the trim should start. This offset is turned into a timecode given the framerate information, and all captions before that timecode are not included in the result CC file.
trimEndExpression STRING	You can trim the CC conversion result, ending at the last caption before the specified trim end. The trimEndExpression must resolve to a double representing the number of seconds into the video the trim should end. This offset is turned into a timecode given the framerate information, and all captions after that timecode are not included in the result CC file.
otherArgsExpression STRING	Other arbitrary MacCaption arguments may be passed into the conversion command. See the MacCaption CLI documentation for available options.
resultDataDef STRING	Stores the converted CC file into a dataDef. The dataDef must be a file.

convertImageStep

The Convert Image step converts an image asset or file into a different image format using ImageMagick.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	Specifies the source image file to convert. The expression must resolve to a file, an image asset, an image asset content, or a string that represents a valid image file path. It can also extract images from JPEG MOV files.

targetImageTemplateExpression STRING	References a target image content template. If specified, the step reads and uses the ImageMagick parameters set in the template.
targetExtensionExpression STRING	An expression that should resolve to a valid image file extension string. Preference is given to the target extension of the image content template resolved by targetImageTemplateExpression; this property should only be set if you are not using a template.
imageMagickParamsExpression STRING	An expression that should resolve to one or more valid ImageMagick parameters. Preference is given to the params property of the image content template resolved by targetImageTemplateExpression; this property should only be set if you are not using a template.
resultDataDef STRING	Stores the converted image file into a dataDef. The dataDef must be a file.

convertVideoStep

The Convert Video step converts a video file or asset into a target format with trimming and pre/postroll options.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression STRING	Specifies the source video file to convert. The expression must resolve to a file, video asset, video asset content, or string that represents a valid video file path. Either this variable or *imageSequenceInfoExpression* must resolve to a not-null value.
imageSequenceInfoExpression STRING	Expression that resolves to an image sequence info JSON structure. Image sequence info structures are normally obtained using the #imageSequenceInfo function on a directory, asset, or asset content. Either this variable or *sourceFileExpression* must resolve to a not-null value.
sourceFrameRateExpression STRING	Specifies the source frame rate. The expression must resolve to a double. This variable is useful for image sequences, where the frame rate cannot be determined.
outputDirectoryExpression STRING	Specifies an output directory for the result of the conversion. The expression must resolve to a directory or string that represents a valid directory.
outputFilenameExpression STRING	Specifies an output filename for the result of the conversion. The expression must resolve to a string that represents the desired filename. Note that if this value's extension is incompatible with the output format of the conversion template, the extension is overridden with the correct format extension.

targetContentTemplateExpression STRING	An expression that represents the target format of the video conversion. The expression must resolve to a media conversion template or a video content template, or the IDs or names for either. As of Reach Engine 9.3, asset content templates are being deprecated in favor of media conversion templates, and workflow XML should use the *mediaConversionTemplateExpression* instead. This attribute will currently find a media conversion template if this expression matches one, but in a future version, this attribute will be fully removed.
mediaConversionTemplateExpression STRING	An expression that represents the target video conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template, you should fully qualify it by specifying the converter type as well as the template name (ex: "Vantage:MyWorkflow").
inputChannelCountExpression STRING	Specifies the number of input channels for channel mapping.
outputChannelCountExpression STRING	Specifies the number of output channels for channel mapping.
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated video. The prerollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path. Note that the preroll video MUST contain an audio track, even if silence, or else the result file's audio will not align correctly.
prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.
postrollFileExpression STRING	Includes a postroll to the generated video. The postrollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.
trimStartExpression STRING	Trims from the start of the main video resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.

trimEndExpression STRING	Trims from the end of the main video resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
captionInjectFileExpression STRING	Includes a .SCC closed captioning file to be injected into the generated video.
paddingTypeExpression STRING	Sets the video padding type to None (Distort), Cut, or Letterbox (Pad).
sidecarAudioExpression STRING	Expression that resolves to a list of sidecar audio files.
conversionJobNameExpression STRING	Expression that resolves to a string that is used to identify the external conversion job, if applicable.
resultDataDef STRING	Stores the converted video file into a dataDef. The dataDef must be a file.

This step contains all common elements, as well as the following step-specific elements.

Elements

conversionParam	Name variable is required.
conversionInfo	Optional converterType variable.

copyFileStep

The Copy File step copies a file or string content to a target directory.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

targetFilenameExpression STRING	Specifies a target filename. Specifying a target filename is required if the sourceFileExpression resolves to string content instead of a file. If sourceFileExpression does resolve to a file, the extension in targetFilenameExpression is ignored, and the source file's extension is used instead.
sourceFileExpression required STRING	Specifies the source file or string content to use. If you are referencing a source file, the expression result can be a string to a qualified file path, a file, XML, JSON, or an asset. If you are referencing content, the expression result must be a string.

targetDirectoryExpression required STRING	Specifies the target directory for the copy. The expression result must be a string to a qualified file path or a directory.
resultDataDef STRING	Stores the copied file in a dataDef. The dataDef must be a file or a string.
createTargetDirectoryFlag BOOLEAN	Defines whether or not to create the target directory if it does not exist or stall the workflow. Setting createTargetDirectoryFlag to false ensures that a mistyped path does not inadvertently get created, but only use this option if the target directory is known and not at all dynamic. Default value is true.
progressPctFrequencyExpression STRING	Specifies how often to update the step's progress. Lower numbers increase granularity at the expense of performance. Default value is 5, which means every 5%.

createCsvStep

The Create CSV step generates a CSV file from the result of a DataObject query with select parts. By using select parts, a tabular result is created.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

targetFilenameExpression STRING	Specifies a target filename. The extension in targetFilenameExpression is ignored; the extension is always .csv.
targetDirectoryExpression required STRING	The target directory for the copy. The expression result must be a string to a qualified file path or a directory.
sourceDataExpression STRING	Specifies a workflow expression that should be used as the source of the CSV as opposed to using the criteria element. The expression must evaluate to one or more DataObjects whose class matches the class specified in the targetDataObjectClass attribute.

This step contains all common elements, as well as the following step-specific elements.

Elements

criteria	

select	At least one instance is required. Select elements declare a DataObject path to select as well as a column label for the results. When the comma separated value file (CSV) is produced, the output columns are in element declaration order. The name of the select element should be the column header name, while the text value should be a valid, persistent DataObject path for the configured target class or an expression rooted on the target DataObject class of the query.

delayStep

The Delay step puts the workflow execution into a WAITING status until the configured delay time/ deadline has been reached. Only one of the delaySecondsExpression or the delayUntilExpression properties should be set.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

delaySecondsExpression STRING	A Reach Engine Expression that should evaluate to an integer number of seconds to delay, or a literal number of seconds to delay.
delayUntilExpression STRING	A Reach Engine Expression that should evaluate to either a date, date/time, or a string that can be parsed into a date/time. If no time is included, the assumption is to delay until midnight of the date specified.

deleteAssetStep

The DeleteAsset step deletes the specified asset from the Reach Engine database and from the repository. It should only be used if you truly want to remove all traces of the asset from the system, such as in the case of a bad import.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

assetExpression STRING	Expression resolving to an asset that will be deleted.
deleteSourceFile STRING	Expression resolving to a Boolean indicating whether to delete the original source file when deleting asset records. Default value is true.

deleteDataObjectStep

The DeleteDataObject step deletes data from the Reach Engine database.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

dataObjectExpression required STRING	Optionally references one or more DataObjects to delete. The expression must resolve to a DataObject (single or multiple).
continueOnException BOOLEAN	Overrides the default value for this step. Default value is true.

deleteFileStep

The Delete File step permanently deletes one or more files and/or directories. Note that this delete is not reversible and so should be used with extreme care. It is normally used to clean up temporary files like transcode artifacts.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFilesExpression required STRING	An expression that should resolve to one or more files and/or directories. The expression must resolve to a file, directory, or a valid string file path; multiple instances of this variable are allowed. All resolved items are deleted.
recurseSubfoldersExpression STRING	If the file being deleted is a folder then this attribute indicates whether to delete its subfolders recursively. Note that if the file being deleted is not a folder then this attribute is ignored. Default value is false.
continueOnException BOOLEAN	Overrides the default value for this step. Default value is true.

emailStep

The Email step produces email text with optional attachments and sends it to the email addresses specified.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

emailAddressesExpression required STRING	An expression that should resolve to one or more valid email addresses. Multiple email addresses can be provided either as a multiple string or as a comma-delimited string of email addresses.
freemarkerRootObjectExpression STRING	If you are using email templates to generate the email body, you can set this expression to change the root context object into Freemarker. By default, the step itself is the root object so standard workflow expression paths will work.
subjectExpression required STRING	Sets the subject of the email. The subjectExpression must resolve to a string.
emailTemplateExpression STRING	Sets the Freemarker email template to use for email body generation. emailTemplateExpression must resolve to a Freemarker template or the valid name of one. Either this variable or the body element must be set.
attachmentsExpression STRING	Attaches the specified file to the email. attachmentsExpression can resolve to an asset, an asset content, a file, or a valid string path to a file. Multiple files can be added to a string by comma-delimiting paths.

This step contains all common elements, as well as the following step-specific element.

Element

body	Specifies the body of the email. Use this element as an alternative to the emailTemplateExpression if you only need a simple email body without any conditional or iterative elements. Use a CDATA tag to keep the formatting of your body intact. The body text can contain expressions.

embedClosedCaptioningStep

If MacCaption/Manzanita integration is enabled in Reach Engine, the Embed Closed Captioning step provides facilities to embed closed captioning into movie files.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceCcFileExpression required STRING	An expression that must resolve to the source closed captioning (CC) file to embed. The expression must resolve to a file, an asset, an asset content, or a string representing a valid file path, and the target file must be a valid closed captioning file.
sourceVideoFileExpression required STRING	An expression that must resolve to the source video file for embedding. The expression must resolve to a file, an asset, an asset content, or a string representing a valid file path, and the target file must be a valid video file.
ccFormatExpression STRING	This expression should evaluate to a known closed captioning format. Currently, only "608" and "708" are supported.
otherArgsExpression STRING	Other arbitrary MacCaption arguments may be passed into the conversion command. See the MacCaption/Manzanita CLI documentation for available options.
resultDataDef STRING	Stores the converted video file into a dataDef. The dataDef must be a file.

enableWatchFolder

The Enable Watch Folder step ensures that there is an enabled watch folder at the specified location and config.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

targetDirectory required STRING	Specifies the directory path to watch. An expression must resolve to a directory or a string that is a valid directory path. Reach Engine creates the directory if necessary.
targetWorkflowKey required STRING	Specifies the workflow that should be run when a new file is detected in the watched directory.
fileDataDef required STRING	The name of the file dataDef in the target workflow that should be set with the detected file.
Subject STRING	Specifies an expression for the subject of the target workflow.
maxConcurrent STRING	Sets the number of workflows that should be able to run concurrently as files are detected in the target directory. Default value is 1.

episodeConvertAudioStep

The Episode Convert Audio step extends the Convert Audio step with Telestream Episode Engine-specific parameters.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	The source audio file to convert. The expression must resolve to a file, an audio asset, an audio asset content, or a string that represents a valid audio file path.
snsEnabledExpression STRING	Specifies whether to enable or disable Episode's Split-n-Stitch processing capability on the conversion. Default value is true.
snsMaxSplitsExpression STRING	If Split-n-Stitch is enabled, optionally specify the maximum number of splits that should be employed by Episode Engine. The expression should resolve to an integer or a valid integer string.
snsMinDurationExpression STRING	If Split-n-Stitch is enabled, optionally specify the minimum duration before it should be employed. In other words, even if Split-n-Stitch is enabled, if the source audio's duration is less than this minimum duration, it will be disabled for the conversion. Testing has shown that if audio is under a certain duration (varies per environment), Split-n-Stitch has a negative performance impact. This expression must resolve to an integer, a double, or a valid decimal string that represents the minimum duration in seconds.
targetContentTemplateExpression STRING	An expression that represents the target format of the audio conversion. The expression must resolve to an audio content template or the name of a valid audio content template.
mediaConversionTemplateExpression STRING	An optional expression that represents the target audio conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template, you should fully qualify it by specifying the converter type as well as the template name (ex: "Vantage:MyWorkflow").
inputChannelCountExpression STRING	Specifies the number of input channels for channel mapping.
outputChannelCountExpression STRING	Specifies the number of output channels for channel mapping.
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated audio. The prerollFileExpression must resolve to an audio asset, an audio file, or a string that represents a valid audio file path.

prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.
postrollFileExpression STRING	Includes a postroll to the generated audio. The postrollFileExpression must resolve to an audio asset, an audio file, or a string that represents a valid audio file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.
trimStartExpression STRING	Trims from the start of the main audio resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main audio that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.
trimEndExpression STRING	Trims from the end of the main audio resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main audio that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
resultDataDef STRING	Optionally stores the converted audio file into a dataDef. The dataDef must be a file.

episodeConvertVideoStep

The Episode Convert Video step extends the Convert Video step with Telestream Episode Engine-specific parameters.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

mediaConverterTypeExpression	
snsEnabledExpression STRING	Specifies whether to enable or disable Episode's Split-n-Stitch processing capability on the conversion. Default value is true.
snsMaxSplitsExpression STRING	If Split-n-Stitch is enabled, optionally specify the maximum number of splits that should be employed by Episode Engine. The expression should resolve to an integer or a valid integer string.

snsMinDurationExpression STRING	If Split-n-Stitch is enabled, optionally specify the minimum video duration before it should be employed. In other words, even if Split-n-Stitch is enabled, if the source audio's duration is less than this minimum duration, it will be disabled for the conversion. Testing has shown that if audio is under a certain duration (varies per environment), Split-n-Stitch has a negative performance impact. This expression must resolve to an integer, a double, or a valid decimal string that represents the minimum duration in seconds.
watermarkExpression STRING	Specifies a Watermark be applied to the resulting video. The expression must resolve to a Reach Engine WatermarkSpec object.
initialTimecodeExpression STRING	Specifies the initial timecode for episode result. This expression should resolve to a double (timecode in seconds from 0).
addQTTimecodeTrackExpression STRING	Specifies whether to enable or disable Episode's "useTimeTrack" option. If not set, the setting in the Episode task is respected.
sourceFileExpression STRING	Specifies the source video file to convert. The expression must resolve to a file, video asset, video asset content, or string that represents a valid video file path. Either this variable or *imageSequenceInfoExpression* must resolve to a not-null value.
imageSequenceInfoExpression STRING	Expression that resolves to an image sequence info JSON structure. Image sequence info structures are normally obtained using the #imageSequenceInfo function on a directory, asset, or asset content. Either this variable or *sourceFileExpression* must resolve to a not-null value.
sourceFrameRateExpression STRING	Specifies the source frame rate. The expression must resolve to a double. This variable is useful for image sequences, where the frame rate cannot be determined.
outputDirectoryExpression STRING	Specifies an output directory for the result of the conversion. The expression must resolve to a directory or string that represents a valid directory.
outputFilenameExpression STRING	Specifies an output filename for the result of the conversion. The expression must resolve to a string that represents the desired filename. Note that if this value's extension is incompatible with the output format of the conversion template, the extension is overridden with the correct format extension.
targetContentTemplateExpression STRING	An expression that represents the target format of the video conversion. The expression must resolve to a media conversion template or a video content template, or the IDs or names for either. As of Reach Engine 9.3, asset content templates are being deprecated in favor of media conversion templates, and workflow XML should use the *mediaConversionTemplateExpression* instead. This attribute will currently find a media conversion template if this expression matches one, but in a future version, this attribute will be fully removed.
mediaConversionTemplateExpression STRING	An expression that represents the target video conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template, you should fully qualify it by specifying the converter type as well as the template name (ex: "Vantage:MyWorkflow").

inputChannelCountExpression STRING	Specifies the number of input channels for channel mapping.
outputChannelCountExpression STRING	Specifies the number of output channels for channel mapping.
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated video. The prerollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path. Note that the preroll video MUST contain an audio track, even if silence, or else the result file's audio will not align correctly.
prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.
postrollFileExpression STRING	Includes a postroll to the generated video. The postrollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.
trimStartExpression STRING	Trims from the start of the main video resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.
trimEndExpression STRING	Trims from the end of the main video resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
captionInjectFileExpression STRING	Includes a .SCC closed captioning file to be injected into the generated video.
paddingTypeExpression STRING	Sets the video padding type to None (Distort), Cut, or Letterbox (Pad).
sidecarAudioExpression STRING	Expression that resolves to a list of sidecar audio files.
conversionJobNameExpression STRING	Expression that resolves to a string that is used to identify the external conversion job, if applicable.
resultDataDef STRING	Stores the converted video file into a dataDef. The dataDef must be a file.

This step contains all common elements, as well as the following step-specific elements.

Elements

conversionParam	Name variable is required.
conversionInfo	Optional converterType variable.

executeSubflowStep

The Execute Subflow step is a foundational Reach Engine Workflow step. It provides workflows with the ability to call other workflows (called subflows) and pass related data down to those subflows. A subflow execution is created for each related data item that is selected. These subflows all execute in parallel, but the execute subflow step waits until all of these subflows complete before continuing to its transitions.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

subjectChangePath STRING	Specifies a path (contextual to the workflow subject) or expression (contextual to the step execution) that should evaluate to one or more elements. Each element is then run through the targeted workflow. By default, these elements are assigned as the subject of the workflow (and thus must be a DataObject) but it is possible to assign them to a contextDataDef instead via the *subflowTargetDataDef* attribute.
subflowTargetDataDef STRING	Specifies the name of a data def to assign the subflow target into as opposed to the subject. Types must match.
emptyTargetHandling	Describes how the step should handle cases where the *subflowTargetsExpression* does not evaluate to any elements. By default, the workflow execution stalls if there are no subflow targets, but you can change this behavior to just complete this step and move on to transitions. Default value is stall.
targetWorkflowId required STRING	The ID of the workflow to run as a subflow.
waitforCompletionExpression STRING	Describes whether the step should wait for subflow completion before continuing. If this resolves to true, the workflow step does not proceed until all subflows are complete. If this resolves to false, the step proceeds immediately. Additionally, if the step is not waiting for completions, subflows that stall or fail to start do not cause this step to stall. Default value is true.

resultDataDef STRING	Specifies the ID of the workflow to run as a subflow.

This step contains all common elements, as well as the following step-specific element.

Element	
subflowContextDataMapping	Subflow context data mappings allow you to equate a contextDataDef in this workflow with a contextDataDef in a subflow, which allows you to pass dataDef values down into a subflow. This element has the following attributes: • parentDataDef : (String) Required. Specifies the name of the dataDef in this workflow. • subflowDataDef : (String) Required. Specifies the name of the dataDef in the target workflow.

failWorkflowStep

The Fail Workflow step puts the workflow execution into a FAILED state. Unlike a stalled workflow, a failed workflow cannot be resumed. Note that you should typically not have any steps after a failWorkflowStep; it should be a terminal step.

This step contains all common attributes and elements, as well as the following step-specific attribute.

Attribute	
reasonExpression required STRING	An optional expression that should detail the reason why the workflow was put into a FAILED state.

getVideoConversionResult

The Get Video Conversion Result step attempts to obtain a result from a video conversion started with the *startVideoConversion* step. Note that if the transcoder/template in use outputs multiple results, each result may be obtained separately with individual *getVideoConversionResult* steps, each with a different *resultKey* attribute value.

If this step is executed and the conversion is not yet completed, the step stays in an EXECUTING state until the conversion is done. The progress of this step tracks along with the progress of the conversion. Additional calls to get other results return immediately since the conversion is complete.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes	
conversionId required STRING	Specifies the conversion ID from which to obtain a result.
resultKey STRING	Specifies a conversion output key associated with this conversion. This attribute is used primarily for transcoders/conversions that output multiple results from a single conversion invocation.
resultDataDef STRING	Stores the output file into a dataDef. The dataDef must be a file.

groovyStep

The groovy step executes a groovy script.

This step contains all common attributes, as well as the following step-specific attributes.

Attribute	
resultDataDef required STRING	Stores the return value of the groovy script.
groovyFilename STRING	Specifies the name of the groovy script. The script must be in the classpath and, therefore, the name given must be a valid resource in the classpath. We suggest that you create the following directory and put your groovy scripts in it: /reachengine/tomcat/lib/groovy If, for example, you had the following groovy script: /reachengine/tomcat/lib/groovy/foo.groovy You would set the 'groovyFilename' equal to 'groovy/foo.groovy'.

This step contains all common elements, as well as the following step-specific element.

Element	
script	This element contains the groovy script. The groovy script must be enclosed within <![CDATA[]]> tags.

identifyMediaStep

A step to return ranked (user defined) Reach Engine media profiles that match the sourceFileExpression

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attribute	
sourceFileExpression required STRING	The source file that will be identified.
offsetExpression optional STRING	Offset the list of media profiles returned. Defaults to 0.
limitExpression optional STRING	Limit the number of media profiles returned. Defaults to 25.

logStep

(Deprecated) A simple step that logs the *outputExpression* result to the Reach Engine log files.

This step contains all common attributes and elements, as well as the following step-specific attribute.

Attribute	
outputExpression required STRING	Specifies the Reach Engine Expression to evaluate and log. This expression can resolve to any data type.

noopStep

No-op steps do not do anything; they simply move to transitions. No-op steps are useful for workflows where the first "real" step must be based upon a transition evaluation. No-op steps are also useful for steps with test conditions and when data must be printed to the UI.

This step contains all common attributes and elements. This step does not contain any step-specific attributes or elements.

Example Usage

```
<noopStep name="check count"
    executionLabelExpression="check count: ${myCount}">
    <transition condition="${ myCount &lt;
formattedOtherDescription.size() }">
        <targetStepName>set field value</targetStepName>
    </transition>
    <transition condition="true">
        <targetStepName>save timeline metadata</targetStepName>
    </transition>
</noopStep>
```

queryStep

The Query step queries the Reach Engine database for data objects.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes	
targetDataObjectClass required STRING	Specifies the target DataObject class to query. Typical classes include asset, clip, timeline, and project. See Reach Engine Metamodel documentation for details.
resultDataDef required STRING	Stores the result of the query into the specified dataDef. The dataDef must be of type Data Object. If the dataDef is multiple, all results are stored; otherwise, only the first result is stored.

This step contains all common elements, as well as the following step-specific element.

Element

criteria	The criteria element is used to produce the search criteria for the query. It follows Reach Engine's Criteria XML format. Criteria strings are normally put into a CDATA block for ease of reading. An example criteria XML is shown below:
	`<![CDATA[`
	`<criteria>`
	`<and>`
	`<condition property="name" op="eq">`
	`<test value="${targetTemplateName}"/>`
	`</condition>`
	`</and>`
	`</criteria>`
	`]]>`

raiseWorkflowEventStep

This step raises a Workflow Event of the type specified. You can add event properties via repeating property elements. See each individual event type for information on what properties are available. Workflow Events are a special way to message users about events that occur during workflow executions. Users can subscribe to event types and configure how they want to be notified when an event occurs, including user interface notification. In this way, they are a more flexible way to message users about workflow events than just sending emails.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

EventTypeExpression required STRING	Specifies the event type name, which is equal to the event's class name. The expression must resolve to a string.
severityExpression STRING	Specifies the severity type. If omitted, the default severity is used. Valid expression result values are "Info", "Warning", and "Critical".
summaryExpression STRING	Specifies the summary expression. If omitted, a default value is used. The expression must resolve to a string.
emailTemplateNameExpression STRING	Specifies the email template to use. If omitted, Reach Engine looks for a template configured for this event type.
emailAddressesExpression STRING	Specifies a set of email addresses to which this event will be sent. If omitted, Reach Engine determines the addresses based on event subscriptions in the system.

This step contains all common elements, as well as the following step-specific element.

Element

property	Specifies an event property value. The *name* attribute is required if this element is used.

readXmpMetadataStep

Need description

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

exifToolPath STRING	Specifies the path to the Exiftool executable.
sourceFileExpression required STRING	Specifies the sourcefile to be submitted to the workflow.
targetObjectExpression required STRING	Specifies the object whose metadata properties should be updated. Can be metadata or a metadata owner.

This step contains all common elements, as well as the following step-specific elements.

Elements

namespacePrefix	Specifies the namespace from which to fetch tags.
property	Specifies which property to fetch.

reviewAndApproveStep

The Review and Approve step emits a resolvable workflow event to users. The user reviews the material attached to the event and either approves or rejects it. The result of the resolution is stored in the resultDataDef variable.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

approvalSubjectExpression required STRING	Specifies an expression that should resolve to a DataObject. This DataObject should be the subject of what is being reviewed.
approvalResultExpression STRING	Specifies an expression that should resolve to a Boolean. This expression can be used instead of a workflow resolution event to complete this step. To use, the expression target must be set with a true/false value and the workflow restarted. The step evaluates this expression and if it result in a not-null value (true or false), it sets the resultDataDef and completes.
Message STRING	Specifies a message to include in the workflow event to help the user know what they should be doing.
resultDataDef STRING	Specifies the name of the contextDataDef that will hold the result of the reject/approve decision. The dataDef must be a Boolean.

rhozetConvertVideoStep

The Rhozet Convert Video step extends the Convert Video step with Rhozet Carbon-specific parameters.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

sourceFileExpression STRING	Specifies the source video file to convert. The expression must resolve to a file, video asset, video asset content, or string that represents a valid video file path. Either this variable or *imageSequenceInfoExpression* must resolve to a not-null value.
imageSequenceInfoExpression STRING	Expression that resolves to an image sequence info JSON structure. Image sequence info structures are normally obtained using the #imageSequenceInfo function on a directory, asset, or asset content. Either this variable or *sourceFileExpression* must resolve to a not-null value.
sourceFrameRateExpression STRING	Specifies the source frame rate. The expression must resolve to a double. This variable is useful for image sequences, where the frame rate cannot be determined.
outputDirectoryExpression STRING	Specifies an output directory for the result of the conversion. The expression must resolve to a directory or string that represents a valid directory.
outputFilenameExpression STRING	Specifies an output filename for the result of the conversion. The expression must resolve to a string that represents the desired filename. Note that if this value's extension is incompatible with the output format of the conversion template, the extension is overridden with the correct format extension.

targetContentTemplateExpression STRING	An expression that represents the target format of the video conversion. The expression must resolve to a media conversion template or a video content template, or the IDs or names for either. As of Reach Engine 9.3, asset content templates are being deprecated in favor of media conversion templates, and workflow XML should use the *mediaConversionTemplateExpression* instead. This attribute will currently find a media conversion template if this expression matches one, but in a future version, this attribute will be fully removed.
mediaConversionTemplateExpression STRING	An optional expression that represents the target video conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template, you should fully qualify it by specifying the converter type as well as the template name (ex: "Vantage:MyWorkflow").
inputChannelCountExpression STRING	Specifies the number of input channels for channel mapping.
outputChannelCountExpression STRING	Specifies the number of output channels for channel mapping.
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated video. The prerollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path. Note that the preroll video MUST contain an audio track, even if silence, or else the result file's audio will not align correctly.
prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.
postrollFileExpression STRING	Includes a postroll to the generated video. The postrollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.
trimStartExpression STRING	Trims from the start of the main video resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.

trimEndExpression STRING	Trims from the end of the main video resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
captionInjectFileExpression STRING	Includes a .SCC closed captioning file to be injected into the generated video.
paddingTypeExpression STRING	Sets the video padding type to None (Distort), Cut, or Letterbox (Pad).
sidecarAudioExpression STRING	Expression that resolves to a list of sidecar audio files.
conversionJobNameExpression STRING	Expression that resolves to a string that is used to identify the external conversion job, if applicable.
resultDataDef STRING	Stores the converted video file into a dataDef. The dataDef must be a file.

This step contains all common elements, as well as the following step-specific elements.

Elements

conversionParam	Name variable is required.
conversionInfo	Optional converterType variable.

runCommandStep

The RunCommand step runs external processes either locally on the host Reach Engine server or remotely via a Reach Engine Remote Exec server.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

executablePathExpression STRING	Specifies an expression that should resolve to a valid path to an executable file located on a Reach Engine server. In order to work across cluster nodes, the path must either be a virtual path or else all servers must have the same path to the executable. The expression must resolve to either a file or a valid path string.
executableNameExpression STRING	An alternative to *executablePath*, executableName must match a configured Remote Exec server command name.

argsExpression STRING	An alternative to *arg* elements, argsExpression is a list of arguments which are separated by whitespace and passed as the arguments to the command. If argsExpression is present, any *arg* elements are ignored. Using argsExpression is useful if the args are not known at the time of creating the workflow step, but instead are coming from user data.
preferRemoteExecution BOOLEAN	Tells the workflow server whether to try to run the command on a remote exec server rather than in the local environment. This attribute can be combined with *remoteHostExpression* to specify a particular remote exec server to use; otherwise, the workflow server selects an available remote exec server. Default value is false.
remoteHostExpression STRING	If specified, the command is run on the specified remote host. remoteHost command executions only work with *executableName* values; executablePath will not be passed to remote hosts for security reasons.
resultCodeDataDef STRING	Specifies a dataDef to store the command result code into.
stallOnErrorResultCode BOOLEAN	Specifies whether or not to stall the workflow if a nonstandard success code is returned. Default value is true, but if you would prefer to ignore or handle error codes manually in transitions then set to false.
stdoutDataDef STRING	Specifies a dataDef to store the stdout result of the command. The dataDef can be of type XML, JSON, or multiple string. If the dataDef is XML or JSON, the output from the command must be valid for the type specified.
stderrDataDef STRING	Specifies a dataDef to store the stderr result of the command. Many external processes write error conditions to stderr instead of stdout. The dataDef must be a multiple string.

This step contains all common elements, as well as the following step-specific element.

Element	
args	Specifies an argument for this command. The text of each arg can be an expression. The arg values are passed into the command in the listed order. All arguments are treated as strings.

saveAssetStep

The Save Asset step creates or updates an asset in Reach Engine's repository.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

contentExpression STRING	Specifies an expression resolving to the location of the binary content to save into the repository. The expression must resolve to either a file or a valid string file path. Either this attribute or *imageSequenceInfoExpression* must be provided for this to be a valid step configuration.
contentFramerateExpression STRING	Specifies an expression that resolves to the framerate of the source material. If omitted, Reach Engine attempts to determine the framerate of the content by examining file header information. The expression must resolve to a double.
imageSequenceInfoExpression STRING	Specifies an expression that resolves to an image sequence info JSON structure. Image sequence info structures are normally obtained using the #imageSequenceInfo function on a directory. To handle multiple sequences, consider writing a top-level workflow that obtains the various sequences and then executes an ingest subflow using the various sequences as the fork point. Either this variable or *contentExpression* must be provided for a valid step configuration.
contentTemplateExpression STRING	Specifies the content template that should be assigned to the saved asset. Content templates provide a "profile" to the content that is stored against an asset. As assets can contain multiple contents, you can save content against an asset with template "Foo" and other content against the same asset with template "Bar", then retrieve each content separately later. It is possible to specify content without a template, but if you try to save the asset again later with other content and no template, the new content will effectively overwrite the previous content. The expression must resolve to a valid string template name or a content template DataObject.
contentUseExpression STRING	Expression that must resolve to a content use string (single or multi). Content uses give Reach Engine information as to how the content should be used in the system. Valid values include Source, Proxy, Edit Proxy, Mezzanine, Thumbnail Video, and Thumbnail.
subjectAssetPathExpression STRING	Deprecated attribute in favor of *assetExpression*, which is more flexible to non-subject asset paths. If you want to update a subject path expression, construct an expression to that asset in the assetExpression value.
assetExpression STRING	Expression resolving to an asset that is used as the target of the asset save.
assetNameExpression STRING	Expression that is used to set the saved asset's name. The name is independent of the filename used for storage. This expression must resolve to a string.
storageNameExpression STRING	Expression that is used to set the saved asset's filename. This expression must resolve to a string.
searchableFlagExpression STRING	Expression that is used to determine whether an asset should be searchable. Generally, assets are not searchable in the UI. An example of assets that we DO want to be searchable are "Other" asset types imported with the "ingestOtherAsset" workflow in Nimbus.
targetRepoExpression STRING	Expression that must resolve to a valid Reach Engine repository name. Repository roots are defined in the Reach Engine properties file using the 'asset.repositoryPath.x' properties, where 'x' is the name of the repository. If omitted, Reach Engine automatically determines the best repository to store the asset.

versionAssetExpression STRING	If an existing asset is specified in assetExpression, versionAssetExpression determines whether or not to create a new version of the asset or update the existing version. This expression must resolve to a Boolean. Default value is true.
resultDataDef STRING	Stores the result of the asset save into a dataDef. The dataDef must be a DataObject.
useMediaInfoExpression STRING	Set to true if using media info. Default value is true
mimeTypeExpression STRING	An expression used to override the mime-type of the given asset.

saveDataObjectStep

The SaveDataObject step saves new or modified data to the Reach Engine database.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

targetDataObjectClass STRING	Specifies the DataObject class to save. This value is only required if saving a new object; if dataObjectExpression resolves to a valid DataObject then *targetDataObjectClass* is ignored. As a best practice, it is a good idea to set this value if you know which type of DataObject you are saving just in case your dataObjectExpression does not yield a result.
dataObjectExpression STRING	Optionally references one or more DataObjects to update, rather than creating a new DataObject. The expression must resolve to a DataObject or collection of DataObjects.
jsonValuesDataDef STRING	Supplies data property values via a JSON dataDef. The JSON data must be a JSON object in the form {name:value, name:value}. Each property name in the JSON object is treated as a DataObject property path and must match a valid DataObject property path to be used. The value must translate to valid data for the given property's data type.
nameValuePairsDataDef STRING	Supplies data property values via a NameValuePair dataDef. For each NameValuePair, the name is treated as a DataObject property path and must match a valid DataObject property path to be used. The value must translate to valid data for the given property's data type.
resultDataDef STRING	Stores the saved DataObject into a dataDef. The dataDef must be of type DataObject.

This step contains all common elements, as well as the following step-specific elements.

Elements

property	Property elements declare certain DataObject property values to set. Each property element should have a name attribute with a valid property name or path for the targetDataObjectClass. The element text should contain a value or expression that is compatible with the property's data type.
jsonData	Specfies DataObject JSON data inline as opposed to using the *jsonValuesDataDef* attribute.

sendFileStep

The Send File step sends any file to a remote location. That remote location can be configured either as a one-off via the *transferTarget* child element or via a preconfigured target using the *transferTargetExpression* attribute.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

sourceFileExpression required STRING	Specifies the source file to send. The expression must resolve to a file, an asset, an asset content, or a string that represents a valid file path.
targetPathExpression STRING	Specifies an output path for the target of the file send. The expression must resolve to a string that represents a valid path for the remote location. If omitted, the file will typically be placed at the root of the transfer target, but the specific behavior is up to the integration.
targetFilenameExpression STRING	Specifies a target filename. The expression must resolve to a string that represents the desired filename. If omitted, the source file name is used.
transferTargetExpression STRING	An expression that represents the transfer target. Transfer targets are configured in Reach Engine ahead of time and can be selected directly by setting the value to something like "S3Target" or $ {#sysconfig('default.s3.target')). Or, you could have the user select the target in the workflow param popup. Either this attribute or the *transferTarget* child element must be populated.
retryCountExpression STRING	Specifies a retry count. The expression must resolve to an integer. If omitted, a default retry count is used, which varies by transport but is usually 1.
resultDataDef STRING	Specifies the name of the contextDataDef that holds the result of the file send. The dataDef must be JSON and is populated with the result of the transfer. The data structure of the result is specified here: TODO provide URL to file transfer JSON structure.

This step contains all common elements, as well as the following step-specific element.

Element	
transferTarget	The transferTarget child element (which is abstract, use a concrete implementation here) allows you to set up transfer target information inline to the workflow rather than having to set up a pre-configured target ahead of time. For example, if you know that a workflow will always be FTPing a file to ftp.example.com with credentials user/password, then you can place a child element inline: `<ftpTransferTarget></ftpTransferTarget>`

setContextData

The Set Context Data step sets the specified contextDataDef to the value generated by *valueExpression*.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes	
targetDataDef required STRING	Specifies the contextDataDef to set. This attribute is not an expression field and the workflow import fails if this data def name cannot be found.
valueExpression required STRING	The result of this expression evaluation is set on the contextDataDef.

startVideoConversion

The Start Video Conversion step launches a video conversion and returns an ID that can be used to look up the result later in the workflow. This step is different from the Convert Video step in that it does not wait for the conversion to complete before marking the step completed. Note that if the transcoder being used is synchronous, this step waits until conversion completes.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

sourceFileExpression STRING	Specifies the source video file to convert. The expression must resolve to a file, video asset, video asset content, or string that represents a valid video file path. Either this variable or *imageSequenceInfoExpression* must resolve to a not-null value.
imageSequenceInfoExpression STRING	Expression that resolves to an image sequence info JSON structure. Image sequence info structures are normally obtained using the #imageSequenceInfo function on a directory, asset, or asset content. Either this variable or *sourceFileExpression* must resolve to a not-null value.
outputDirectoryExpression STRING	Specifies an output directory for the result of the conversion. The expression must resolve to a directory or string that represents a valid directory.
outputFilenameExpression STRING	Specifies an output filename for the result of the conversion. The expression must resolve to a string that represents the desired filename. Note that if this value's extension is incompatible with the output format of the conversion template, the extension is overridden with the correct format extension.
mediaConversionTemplateExpression STRING	An expression that represents the target video conversion template. The expression must resolve to a media conversion template or the name of a valid media conversion template. As a best practice, if specifying the name of a conversion template, you should fully qualify it by specifying the converter type as well as the template name (ex: "Vantage:MyWorkflow").
channelMappingExpression STRING	Specifies channel mappings in the form in=out, where 'in' is the input channel number and 'out' is the output channel number.
prerollFileExpression STRING	Includes a preroll to the generated video. The prerollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path. Note that the preroll video MUST contain an audio track, even if silence, or else the result file's audio will not align correctly.
prerollDurationExpression STRING	Provides a duration of the preroll file specified in prerollFileExpression in decimal seconds. This value is ignored if prerollFileExpression is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the preroll file, the entire preroll file is prepended.
postrollFileExpression STRING	Includes a postroll to the generated video. The postrollFileExpression must resolve to a video asset, a video file, or a string that represents a valid video file path.
postrollDurationExpression STRING	Provides a duration of the postroll file specified in postrollFileExpostssion in decimal seconds. This value is ignored if postrollFileExpostssion is omitted. The value must resolve to an integer, double, or valid decimal string. If the resolved value is longer than the duration of the postroll file, the entire postroll file is appended.

trimStartExpression STRING	Trims from the start of the main video resolved from sourceFileExpression. trimStartExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds into the main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming occurs before the preroll is appended, so prerolls are unaffected by trims.
trimEndExpression STRING	Trims from the end of the main video resolved from sourceFileExpression. trimEndExpression must resolve to an integer, double, or valid decimal string, and represents the decimal seconds from the end of main video that should be trimmed out. Note that if both trimming and preroll values are specified, the trimming will occur before the preroll is appended, so prerolls are unaffected by trims.
resultDataDef STRING	Stores the conversion ID into a dataDef. The dataDef must be a string.

This step contains all common elements, as well as the following step-specific element.

Element

conversionInfo	Need description

subflowContextDataMapping

Subflow context data mappings allow for other data defs to be passed down to the subflows. The data types of the two data defs must match.

This step only contains the following attributes and does not contain any common attributes. Required attributes display as bold and are marked as required.

Attributes

parentDataDef required STRING	Specifies the name of the dataDef in this workflow.
subflowDataDef required STRING	Specifies the name of the dataDef in the target workflow to map data into.

submitHttpStep

The SubmitHTTP step submits an HTTP request with the configured properties and saves the results received.

This step contains all common attributes, as well as the following step-specific attributes.

Attributes

urlExpression STRING	Specifies the URL to invoke for this HTTP request. Alternatively, you can use the *urlExpressionElement*.
requestMethodExpression STRING	Specifies the request method to use. Valid expression result values are "GET", "POST", "PUT", and "DELETE".
responseCodeDataDef STRING	Specifies a dataDef to store the HTTP response code. This data def must be of type string or integer.
responsePayloadDataDef STRING	Specifies a dataDef to store the HTTP response payload. This data def must be of type string, file (usually for binary content), XML, or JSON.
responseHeadersDataDef STRING	Specifies a dataDef to store the HTTP response headers into. This data def must be of type NameValuePair with multiple set to true.

This step contains all common elements, as well as the following step-specific elements.

Elements

requestPayloadItem	Specifies a request payload item. Request payloads are applicable for POST and PUT requests. Name attribute is required.
param	Specifies a request parameter. Request parameters are applicable for GET, POST, and PUT requests. Name attribute is required.
requestHeader	Specifies an HTTP request header to add or set to the request. Request headers are valid for all request types. Both standard (e.g., Content-Type) and custom headers are accepted. Name attribute is required.
urlExpressionElement	An alternative way to specify the URL for this request, the other way being the *urlExpression* attribute. The urlExpressionElement is used if both are specified.

testStep

This step has been deprecated. Do not use.

updateDataObjectCalculatedPropertiesStep

The UpdateDataObjectCalculatedProperties step triggers an update of a DataObject's calculated properties.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attributes

dataObjectExpression required STRING	References a DataObject to update. The expression must resolve to a DataObject.
triggerModification BOOLEAN	Some DataObject calculated properties are only updated after a DataObject has been modified. Set this attribute to 'true' to trigger those properties to be recalculated.

validateMediaStep

A step to validate a sourceFileExpression against a specific (user defined) Reach Engine media profile.

This step contains all common attributes and elements, as well as the following step-specific attributes.

Attribute

sourceFileExpression required STRING	The source file that will be identified.
mediaProfileIdExpression required STRING	Evaluates to the id of the media profile to use for validation.
resultDataDef required STRING	JSON to store the result.

validateXmlStep

Validates an XML file's syntax for well-formed-ness. An XML file can be passed in as a file, absolute path, or XML (in string or XML format).

This step contains all common attributes and elements, as well as the following step-specific attribute.

Attribute	
xmlExpression	